Weight Decay

Ki Hyun Kim

nlp.with.deep.learning@gmail.com

Fast campus

Weight Decay

- L2 Norm을 통해 weight parameter가 원점에서 멀어지는 것을 방지!
  - Bias는 penalty 대상에서 제외!

$$\tilde{\mathcal{L}}(\theta) = \mathcal{L}(\theta) + \alpha \|W\|_2^2$$
$$= \mathcal{L}(\theta) + \alpha W^T \cdot W,$$

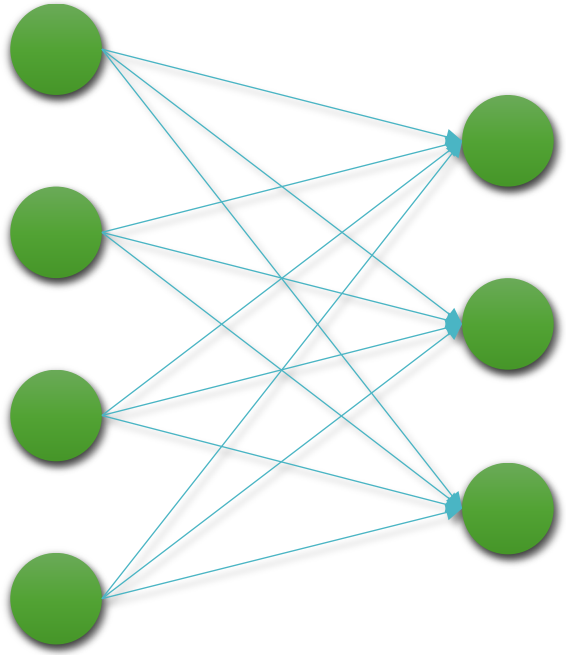
$$\text{where } \theta = \{W, b\}.$$

Fast campus

Why?

- Weight parameter는 노드와 노드 간의 관계를 나타냄
  - 숫자가 커질수록 강한 관계

- 전체적인 관계의 강도를 제한하여
  출력 노드가 다수의 입력 노드로부터 많이 배우지 않도록 제한



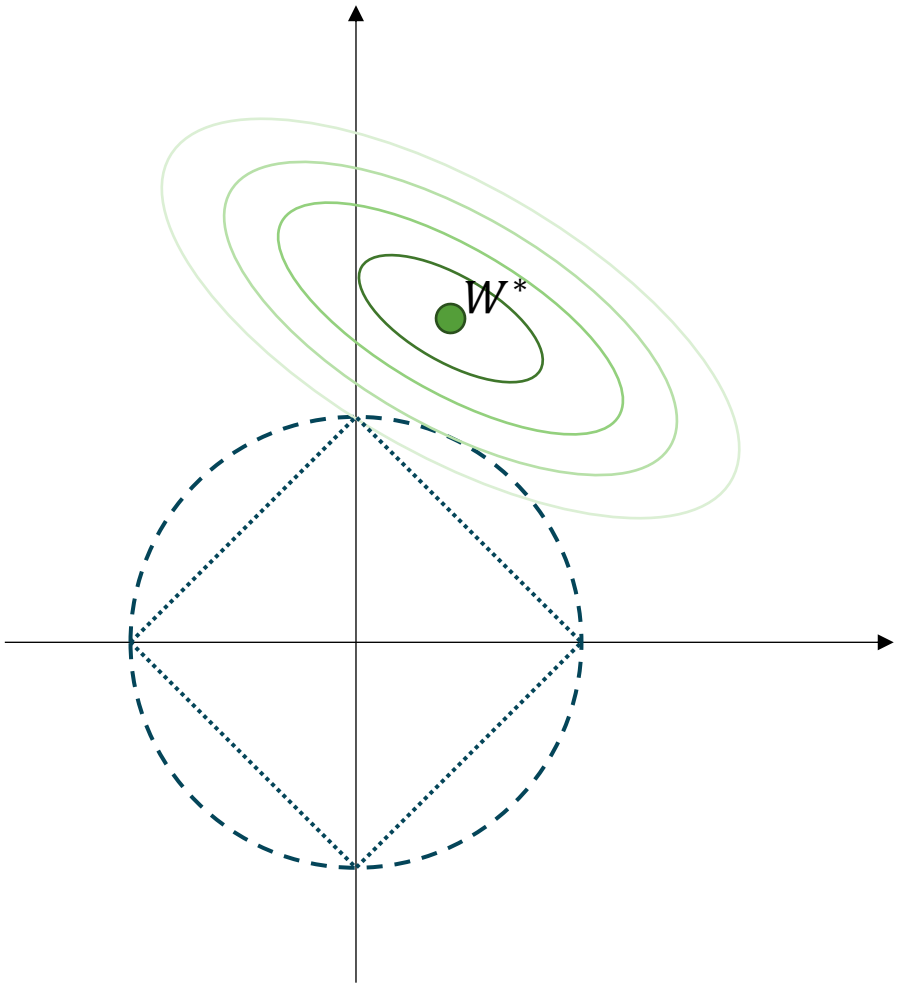
Fast campus

Weight Decay using L1 Norm

- 노드 사이의 관계가 희소(sparse)하도록 제한
  - 더 적은 수의 입력 노드로부터 학습하도록 제한

$$\tilde{\mathcal{L}}(\theta) = \mathcal{L}(\theta) + \alpha \|W\|_1,$$

$$\text{where } \theta = \{W, b\}.$$



Fast campus

Original Loss Minimization을 방해

- Loss 함수에 최소화를 방해하는 term을 넣음
  - Loss가 최소화 될 수록, 커지는 term.

- Hyper-parameter α 를 통해 두 term 사이의 균형을 조절

minimize maximize

$$\tilde{\mathcal{L}}(\theta) = \mathcal{L}(\theta) + \alpha \|W\|_2^2$$
$$= \mathcal{L}(\theta) + \alpha W^T \cdot W,$$

$$\text{where } \theta = \{W, b\}.$$

Fast campus

In PyTorch

- SGD나 Adam 의 초기화 파라미터로 제공

CLASS `torch.optim.SGD(params, lr=<required parameter>, momentum=0, dampening=0, weight_decay=0, nesterov=False)` [SOURCE]

Implements stochastic gradient descent (optionally with momentum).

Nesterov momentum is based on the formula from On the importance of initialization and momentum in deep learning.

Parameters

- **params** (*iterable*) – iterable of parameters to optimize or dicts defining parameter groups
- **lr** (*python:float*) – learning rate
- **momentum** (*python:float, optional*) – momentum factor (default: 0)
- **weight_decay** (*python:float, optional*) – weight decay (L2 penalty) (default: 0)
- **dampening** (*python:float, optional*) – dampening for momentum (default: 0)
- **nesterov** (*bool, optional*) – enables Nesterov momentum (default: False)

Wrap-up

- Weight Decay는 Loss 함수 수정을 통한 regularization 방식의 대표주자
  - Original objective term과 반대로 최적화 되는 regularization term
  - 두 term을 동시에 최소를 만드는 과정에서 overfitting을 방지
  - 두 term 사이의 균형을 유지하는 것이 관건: hyper-paramter를 통해 조절

minimize maximize

$$\tilde{\mathcal{L}}(\theta) = \mathcal{L}(\theta) + \alpha \|W\|_2^2$$
$$= \mathcal{L}(\theta) + \alpha W^T \cdot W,$$

$$\text{where } \theta = \{W, b\}.$$

Fast campus